TAILORING DOD-STD-2167A - A SURVEY OF CURRENT USAGE

A.P. GABB, P.C. POLLARD, S.F. LANDHERR and R.J. VERNIK

COMBAT SYSTEMS DIVISION
WEAPONS SYSTEMS RESEARCH LABORATORY

DTIC
S ELECTE
FEB 26 1992
D
D

Approved for Public Release

OCTOBER 1991

DSTO ▲
AUSTRALIA
DEPARTMENT OF DEFENCE
DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION
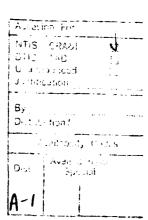
92-04502

92 2 20 039

UNCLASSIFIED

DSTO
AUSTRALIA

TECHNICAL NOTE
WSRL-TN-57/91

TAILORING DOD-STD-2167A -
A SURVEY OF CURRENT USAGE

A.P. Gabb, P.C. Pollard, S.F. Landherr* and R.J. Vernik*

Accession For

NTIS CRA&I

DTIC TAB

Unannounced

Justification

By

Distribution/

Availability Codes

Dist | Avail and/or
Special

A-1

S U M M A R Y (U)

As part of a study into the tailoring of DOD-STD-2167A, a survey was conducted
into the use and tailoring of the standard in Australia. This paper summarises the
results of the survey and makes recommendations for the development of tailoring
guidelines.

Authors' address: *
    Combat Systems Division   Information Technology Division
    Weapons Systems Research Laboratory   Electronics Research Laboratory
    PO Box 1700   PO Box 1600
    Salisbury, 5108   Salisbury, 5108
    South Australia   South Australia

Requests to: Chief, Combat Systems Division

UNCLASSIFIED

TABLE OF CONTENTS

LIST OF APPENDICES

ABBREVIATIONS

1521B	MIL-STD-1521B
2167A	DOD-STD-2167A
490A	MIL-STD-490A
CASE	Computer aided software engineering
CDR	Critical Design Review (1521B)
CDRL	Contract Data Requirements List (490A)
CRISD	Computer Resources Integrated Support Document (DID)
CSC	Computer Software Component (2167A)
CSCI	Computer Software Configuration Item (2167A)
CSOM	Computer System Operator's Manual (DID)
CSU	Computer Software Unit (2167A)
DID	Data Item Description - specification of a document required by DOD-STD-2167A
DSTO	Defence Science and Technology Organisation
ECP	Engineering Change Proposal (2167A)
FQT	Formal qualification testing (2167A)
HWCI	Hardware Configuration Item (2167A)
HQADF	Headquarters, Australian Defence Force
IDD	Interface Design Document (DID)
IRS	Interface Requirements Specification (DID)
OO	Object oriented
OOD	Object oriented design
OOP	Object oriented programming
PDR	Preliminary Design Review (1521B)
QA	Quality assurance
SDD	Software Design Document (DID)
SDF	Software Development Files (2167A)
SDP	Software Development Plan (DID)
SRS	Software Requirements Specification (DID)
SSDD	System/Segment Design Document (DID)
SUM	Software User's Manual (DID)

1. INTRODUCTION

1.1 Purpose

DOD-STD-2167A is a military standard establishing requirements for software development. Much of the operational software currently being developed for the Australian Department of Defence is being developed in accordance with this standard; its use is likely to be mandatory in almost all future operational systems.

The transition to this new standard has not been straightforward. There have been numerous criticisms about DOD-STD-2167A (and its precursor DOD-STD-2167), particularly with regard to the amount of documentation required and the effort needed to produce it. Doubts have also been raised as to the actual value of such documentation to the customer or developer.

The standard claims that it is designed to be tailored for each contract, but it has become apparent that there is some uncertainty both in the Department of Defence and in the commercial software industry about the process of tailoring 2167A for a specific project.

A study is being conducted to investigate the requirements for the tailoring of 2167A in Defence projects. The study is a joint undertaking of Combat Systems Division and Information Technology Division in the Australian Defence Science and Technology Organisation.

This paper describes the first phase of the study - a survey into the use and tailoring of 2167A in Australia. In the second phase, recommendations will be made with regard to tailoring of the standard for Defence projects.

It should be stressed that many of the opinions expressed in this paper are those of participants in the survey, and are not necessarily shared by the authors. In some cases opinions which may not be generally supportable have been included both for completeness and to indicate the different perspectives which the authors encountered.

1.2 Nomenclature

The terms "customer" and "developer" are used in this paper to indicate those responsible for software system procurement and development respectively. In many cases the terms are used to indicate individuals in the customer and development teams rather than the organisation that they represent.

1.3 Scope

Although the study is aimed at addressing the tailoring of 2167A, it will also briefly address other aspects of 2167A usage which are considered important for current and future projects.

1.4 Organisation of this report

Section 2 describes the manner in which the survey was conducted. It also summarises general information gained from the survey relating to development methods, tools, use of metrics and training.

Sections 3 to 6 describe the findings of the survey divided into general DOD-STD-2167A experiences, documentation, development and production issues.

Section 7 addresses specific findings on tailoring and provides an assessment of current tailoring guidelines, tools and training courses.

Sections 8 and 9 summarise the findings of the survey and suggest a way ahead for the development of tailoring guidelines.

The appendices show the original request soliciting interest in the study and the questionnaire used as a basis for interviews.

1.5 Acknowledgments

The authors wish to thank all those in industry, academic institutions and the Department of Defence who have given their time and assistance in contributing to this study.

1.6 Applicable documents

The following standards are referred to in this document.

Australian Standards

AS 3563-1988	Software Quality Management System
AS 3901-1987	(ISO 9001-1987) Quality Systems for Design/- Development, Production, Installation and Servicing

Military Standards

MIL-HDBK-287	Tailoring Guide for DOD-STD-2167A
MIL-STD-483A	Configuration Management Practices for Systems, Equipment, Munitions and Computer Programs
MIL-STD-490A	Specification Practices
MIL-STD-499	Engineering Management
MIL-STD-1467	Software Support Environment
MIL-STD-1521B	Technical Reviews and Audits for Systems, Equipments and Computer Software
DOD-STD-2167A	Defense System Software Development
DOD-STD-2168	Software Quality Program

Data Item Descriptions (DIDs)

DI-CMAN-80534	System/Segment Design Document
DI-MCCR-80030A	Software Development Plan
DI-MCCR-80025A	Software Requirements Specification
DI-MCCR-80026A	Interface Requirements Specification
DI-MCCR-80027A	Interface Design Document
DI-MCCR-80012A	Software Design Document
DI-MCCR-80029A	Software Product Specification
DI-MCCR-80013A	Version Description Document
DI-MCCR-80014A	Software Test Plan
DI-MCCR-80015A	Software Test Description

DI-MCCR-80017A	Software Test Report
DI-MCCR-80018A	Computer System Operator's Manual
DI-MCCR-80019A	Software User's Manual
DI-MCCR-80021A	Software Programmer's Manual
DI-MCCR-80024A	Computer Resources Integrated Support Document

2. OVERVIEW OF THE SURVEY AND PARTICIPANTS

As part of the study the authors conducted a survey of 2167A policy and usage in software development projects. Although the study is primarily aimed at defence projects, there are several non-defence applications of the standard in Australia, both in internal developments and in commercial applications. These were also considered in the survey.

This section addresses the manner in which the survey was carried out, followed by general information gained from the survey relating to development methods, tools, use of metrics and training.

2.1 Canvassing interest

An initial letter soliciting interest was sent to a wide range of addressees (Appendix I). The aim of the mailing list was to contact as many parties in Australia as possible having experience or interest in the use of 2167A. These included policy, project and research areas in the Department of Defence, software developers in industry and academic institutions.

More than half of those contacted replied positively, either offering assistance or, where the respondent had little or no 2167A experience, expressing interest in the results of the study and wishing to be kept informed. Since the initial mailing the authors have been contacted by others who were accidentally omitted and wished to participate. In a few cases the authors specifically requested interviews with organisations which either had not responded or had been omitted from the initial mailing.

2.2 Participants

In total 34 participants were interviewed. These included most of the major defence software developers in Australia, as well as software policy areas and major projects using DOD-STD-2167A in the Department of Defence. The general breakdown of participants is shown in the table below.

Project staff (including consultants such as DSTO)	10
Developers (including 2 Defence software development houses)	15
Policy areas (representing Navy, Air Force, Army, DSTO and HQADF)	6
Other (training and academic institutions)	3

22 projects were discussed of which 18 are being developed to 2167A and the remainder to its predecessor DOD-STD-2167. Of the 2167A projects, 5 are at too early a stage for any serious development experience to have been gained from them.

In a few of the larger projects there are separate software developments by different contractors, in at least one case to different development standards.

Almost all of the developments are for operational real-time sensor, weapon or command and control systems. The remainder are support systems for the operational systems and include test, simulation and information systems with multiple on-line terminals.

2.3 Interviews

Prior to interviews a questionnaire was sent to participants (Appendix II), allowing them to consider responses in advance. In a few cases participants provided written responses to the questionnaire, although this was not specifically requested.

As can be seen from the questionnaire, participants were guaranteed confidentiality in the details of the interviews. Although this approach reduces the impact of a report such as this, by not being able to quote examples directly, the authors believe that it was justified by the frankness of the consequent discussions. Some of the comments made were in fact unprintable. It also allowed staff to provide personal, as opposed to corporate, opinions.

Each interview was of one to three hours duration, depending on the experience of the participant with 2167A, among other factors.

Some participants were also influenced by previous experiences with 2167A's precursor DOD-STD-2167. The authors recognise the fact that 2167A is significantly different both in general and documentation requirements and have taken this into consideration in interpreting and filtering participants' opinions.

2.4 Workshops

Following distribution of the draft of this report to interested participants, half-day workshops were held in Canberra and Adelaide to discuss problems in using 2167A. Each workshop was attended by approximately 30 people. This final report includes issues raised in these workshops.

2.5 DOD-STD-2167A Projects

Although there are several projects currently being developed in accordance to 2167A, the authors did not encounter any which might be considered complete. There are sufficient experiences with projects in progress, in the authors' opinion, to assess most of the problems being encountered, but the full effect of these problems is yet to be seen.

Wherever possible, a project's customers and developers were interviewed independently to gain a perspective of the real impact of 2167A on the project. The differences of opinion were usually highly illuminating.

2.6 Development methods

All developers claimed to have a consistent method for analysis and design, although few have internal written guidelines or standards for these tasks. Most methods used are based on popular textbooks and include:

(a) Structured Analysis / Structured Design based on Yourdon, De Marco, Ward-Mellor et al.

(b) Object oriented methods based on Coad, Yourdon et al.

(c) Ada specific methods, such as Booch's "Software Engineering in Ada".

2.7 CASE tools

Although most developers appeared satisfied with their selection of analysis and design methods, there is almost universal dissatisfaction with the corresponding CASE tools. In many cases, the tools available are used only for analysis, and not for design, and in some instances "only as a drawing tool". Tools used are as follows (in approximately decreasing order of use):

Teamwork (Cadre)
Software Through Pictures (IDE)
Excelerator (Index)
AdaGen (Mark V)
Softbench (Hewlett Packard)
Internally developed

Criticisms of the CASE tools used generally fell into one or more of the following categories:

(a) The tools do not adequately match the methods used (particularly object oriented methods).

(b) The drawing capability is adequate for the original analysis and design, but is poor for making changes.

(c) The tools are difficult to incorporate into established documentation processes and difficult to customise for specific developer requirements.

(d) The tools are barely adequate for analysis, and a waste of time for design.

Most of the developers are considering changing to different tools (with which other developers are also currently dissatisfied) indicating that the CASE tools currently provided for analysis and design are far from optimal.

2.8 Use of metrics

The use of metrics for process control and estimation appears to be relatively limited. Although many developers collect metrics on design and development activities, few use this data for any specific purpose. The typical attitude is: "We are collecting metrics, but haven't decided how (or do not have enough time) to use them yet".

In several cases developers use metrics tools for complexity analysis and for other simple tasks (eg counting lines of code, calculating code/comment ratios).

2.9 Training

Few of the developers interviewed follow a comprehensive formal training program for their staff. Although developers generally acknowledged the need for formal training, it seems that most staff learn from on-the-job training and attendance at the occasional seminar (particularly those organised by Technology Training Corporation). General training of staff in the application of DOD-STD-2167A, either in internal or external courses, appears to be rare.

Ada developers indicated a more structured approach to design and language training, and this is probably indicative both of the scarcity of recruits with Ada experience and the additional discipline required in designing Ada systems.

Several developers commented on the lack of preparedness of new graduate recruits for work in a disciplined software engineering environment, and were critical of Computing Science courses in this regard.

Several developers, customers and Defence policy makers had attended the seminar on 2167A tailoring provided by Technology Australia. This is discussed further in Section 7.2.3.

3. DOD-STD-2167A - GENERAL ISSUES

This section addresses general issues related to DOD-STD-2167A.

3.1 General perspectives

Most of those interviewed agreed that 2167A is a reasonable standard when considered separately from the Data Item Descriptions (DIDs), and direct criticism of the DIDs came almost exclusively from developers. Several participants regarded it as the best structured and best written of the major military specifications that they work with.

Smaller software developers, with 30 or less staff directly involved in the development of software, have an understandable difficulty in meeting the full requirements of the standard, particularly with regard to providing genuinely independent quality assurance (QA) and unit testing teams.

Developers and customers agreed that without the appropriate training and experience on both sides, following 2167A could lead to serious problems in the project. Both developers and customers need to understand the process required by the standard and appreciate the need for tailoring. Both also need to be prepared for the enforced discipline which is likely to be stricter than with other agreed standards.

The was a general feeling that more guidance is required in the meaning of and rationale for some of the clauses in 2167A. Its terseness was seen as one of the causes of problems when disagreements arise between developers and customers as to interpretation of the standard. The relative scarcity of books and articles relating to 2167A development was also seen as a problem.

3.2 Customers' perspectives

Many customers' complaints related to program schedule delays and the inadequacy of delivered documentation. In more than one case these were intertwined: the developer blamed 2167A for his inability to produce review documentation on time, while the customer felt that the developer's inexperience with the standard was the main cause.

Some customers also suggested that developers tend to adopt a "document-driven" approach to 2167A development - that the main areas of the standard are not understood or followed and that the only objective of the developer is to provide the required documentation. In a similar vein it was suggested that some customers believe that the thickness of the delivered documents is as important as the content in gaining customer approval.

Many customers were dissatisfied with the content and quality of documents delivered for design reviews. A common criticism that too much of the wrong type of data is delivered.

There were several customers who believed that 2167A should provide more control than it does. Perceived omissions included:

(a) 2167A does not adequately cover the full development life cycle.

(b) The standard should include a defined design method.

(c) Insufficient guidance is given to the negative aspects such as what action is to be taken when documents are not approved (particularly when used in conjunction with MIL-STD-1521B).

(d) The DIDs are not sufficiently explicit to guarantee good documentation.

Although these views were relatively isolated (apart from the first, and 2167A does not claim to cover other than initial development), they indicate a willingness to blame the standard for problems that are incorrectly or inadequately addressed elsewhere. With regard to these comments:

- 2167A intentionally leaves the choice of the analysis and design methods open, which is not surprising considering that there is little agreement in the industry as to the best approach for different applications.

- Actions to be taken when agreement cannot be reached depend very much on the particular project and need to be detailed in the Statement of Work or contract.

- It is generally agreed that "good" documentation cannot be achieved by the imposition of standards alone.

3.3 Developers' perspectives

Developers' complaints were mainly concerned with the lack of experience of their customers and the need for tailoring of the standard. Several developers considered that

their customers did not have sufficient experience or training in software engineering or the use of 2167A. Specific comments included:

(a) Customers tend to read the clauses too literally, particularly with regard to the DIDs.

(b) Customers do not understand the usefulness of tailoring to them (the customers) and do not have the experience to approve a reasonable tailoring.

(c) There is strong resistance to any except the most minor tailorings.

(d) Tailoring is seen by the customer as an attempt by the developer to reduce the customer's visibility of the development.

The problem here appears to be a combination of distrust and inexperience on the part of the customer. In defence of their cynicism, customers cited tailorings proposed by developers which were subsequently rejected. Examples included proposals that:

• the requirements of 2167A would be tailored by replacing them by internal company standards (where written standards did not exist), and

• the SDD would not be provided because equivalent information would be provided in the source listings.

3.4 The effect of changes in 2167A projects

One criticism that is sometimes levelled at 2167A is that because it is documentation intensive, changes introduced during the development cycle are overly expensive. This argument may then be extended to the proposition that using 2167A stifles necessary changes.

There was general consensus that changes are expensive in any disciplined and properly managed software development, and that the later the changes are incorporated the greater the cost impact will be. While some developers consider that 2167A provides few additional impediments to change, there were several comments regarding specific problems in using the standard.

(a) If there is too much detail in the higher level documents (such as the SSDD) small changes can affect too many documents.

(b) A similar effect can occur if the user requirements are over-specified, and include specific solutions. In this situation there will be a tendency for design details to be introduced in the SSDD and SRS documents.

(c) If there is significant duplication in documents, such as can occur with insufficient tailoring in relatively small projects, changes are likely to be missed in some documents, leading to inconsistencies. Tailoring needs to consider the effects of changes.

3.5 Compatibility of 2167A with other standards

We discussed the compatibility of other standards with 2167A. These included:

MIL-STD-490A	Specification practices
MIL-STD-499	Engineering management
MIL-STD-483A	Configuration management
MIL-STD-1521B	Technical reviews and audits
DOD-STD-2168	Software quality
AS 3563-1988	Software quality
AS 3901-1987	(ISO 9001-1987) Quality systems

There was a wide disparity of opinion on this subject. Many considered that if the recommendations of MIL-HDBK-287 are heeded (see Section 7.2.1), that there are no serious incompatibilities. Others indicated that they saw serious problems with overlaps and conflicts, particularly with 1521B.

A possible reason for this dichotomy is that in many projects the requirements of 1521B are not strictly enforced, either as a result of contractual agreement or by common understanding. Consequently, many developers (and customers) are unaware of the full force of the standard.

Many participants saw few problems with the interaction between hardware and software development standards. Moreover, there appeared to be a willingness to isolate software development from the hardware development process where possible. Other developers considered that, because of the overlap between standards such as 2167A, 499, 490A and 1521B, the decision on which parts of the relevant standards would be used could seriously affect the smooth running of the project.

One area of common agreement is that 1521B is not directly compatible with the incremental or spiral development models. This is discussed further in Section 5.1.

3.6 The use of 2167A in different types of projects

Most of the projects in which the survey participants were involved were for real-time applications. Several developers considered, however, that the rigorous requirements of 2167A are too stringent for the development of information systems, particularly with regard to analysis and design documentation, and that tailoring is essential.

4. DOCUMENTATION ISSUES

4.1 Data Item Descriptions (DIDs)

Much of the criticism of 2167A is directed at the Data Item Descriptions (DIDs) which define the format and content of the delivered documentation.

4.1.1 General comments

General criticisms of the DIDs were as follows:

(a) The DIDs require too much detail, and take too much effort to prepare (developers).

(b) The DIDs do not guarantee the correct level of visibility (customers).

(c) It is too easy for a developer to produce documents of dubious quality - quality of documentation should be defined in the DIDs (customers).

(d) The interpretation of the DID requirements can vary widely among different developers and customers - the requirements should be more carefully explained.

(e) The DID formats are strongly based on the waterfall development model, and cannot be used untailored for some other methods, particularly object oriented methods.

(f) The DID formats are inappropriate for the documentation of some applications, and do not provide for the inclusion of additional information that might be necessary.

(g) The DID specification formats are unsuitable for the expression of operational and functional requirements, where these must be validated by users who do not possess software engineering experience.

Several of these comments are addressed in Section 4.2.

Several developers considered that preparation of documentation would be much easier if the customer could provide examples of the various DID documents, or define precisely what is required in each document.

The level of detailed required in documentation, and the subsequent cost, should vary with different projects, and hence is a tailoring issue. It is evident that customers, often with experience of documentation developed to "company internal standards" are usually prepared to meet that cost.

The authors agree that significant tailoring is necessary to meet some modern development methods (see Section 5.1). Tailoring may also be necessary to suit particular applications. Most participants agreed that some of the problems suggested with inappropriate DID formats can be overcome with little or no tailoring by the use of annexes providing the required data in a suitable format.

It is unlikely that tailoring will solve the problems of requirements expression and validation. Education of the users in the methods used for stating requirements was suggested as a partial solution; another was the extraction of sections of 2167A documents into annexes (possibly as separate documents) where the format may be more appropriate to the application.

4.1.2 Specific comments

Comments on specific DIDs were as follows:

(a) There is too much duplication between the IRS, IDD and SRS with regard to interface data.

(b) For a single CSCI, interface data should be only in the SRS - there is no need for the IRS or IDD.

(c) The IDD is not necessary for Ada developments.

(d) There is no obvious place in the DID structure for the definition of the user interface. While it is a design (and perhaps an interface design) it is also a requirement for the software design, and should be defined at the Preliminary Design Review (PDR) according to MIL-STD-1521B.

(e) The CSOM and SUM manuals are not in a suitable format for many applications.

(f) The CRISD requirements are inadequate - it should be tailored to include the requirements of DOD-STD-1467.

(g) Object oriented methods obviate the need for an IDD because the interface can be encapsulated in an object.

These views appear to have been made from experience with only one or two projects. While the authors cannot agree with all of these comments for all projects, it would appear that all (except perhaps the last) are valid in some circumstances.

4.2 "Good" documentation

There appears to be an almost universal difference of opinion between developers and customers regarding the suitability of delivered documentation - at PDR, CDR and final delivery. The authors were particularly interested in participants' views on how the customer's aim of "good" documentation can be achieved.

Developers were almost unanimous in suggesting that the only way in which a customer can get the standard of documentation he requires is for the customer and developer to have a much closer working relationship, particularly in the early stages of the project. There was some doubt expressed, however, about the customer's ability to define the standard and quality of documentation required. A few developers also suggested that their customers' expectations were too high.

Although closer working relationships were also suggested by some customers as leading to better documentation, several also proposed more stringent standards controlling the quality of documentation. Some also suggested that the only way to guarantee reasonable design documentation for software maintenance is for the customer's staff to write it.

5. DEVELOPMENT ISSUES

This section addresses the development issues in 2167A projects, with emphasis on the analysis and design phases and the production of the following documents:

SDP	Software Development Plan
SSDD	System/Segment Design Document
IRS	Interface Requirements Specification
IDD	Interface Design Document
SRS	Software Requirements Specification
SDD	Software Design Document

5.1 The relationship of 2167A to the software development method

All developers were concerned about the compatibility of 2167A with current and evolving development methods. Most considered that while 2167A does not specify a software development method, much of the standard is based on the waterfall model, and needs significant tailoring to be adapted to other methods such as the incremental, evolutionary, spiral and object oriented models (and combinations of these).

It was agreed as important that if the development method diverges significantly from the waterfall method, this should be addressed in detail in the Software Development Plan (SDP), with particular regard to the control of the development process and documentation.

5.1.1 Incremental development

The incremental development approach is based on the concept of phased development. Successive versions (or builds) include additional functionality. Each version provides a useable subset of the required functions and performance (as specified in a baselined requirement), and may be delivered to the customer. The deliverables of incremental development are often final constituents of the final product.

The main concerns are as follows:

(a) It is unlikely that successive builds can be planned such that changes to the design documentation for CSCIs already delivered can be avoided. Previously approved and delivered IRS, IDD, SRS and SDD documents, as well as test documentation may need to change between versions. This is seen as a major cost to the development effort, and a serious problem in configuration control.

(b) 2167A documentation requires completeness with regard to design and requirements traceability. With an incremental approach to development, design of subsequent versions is usually deferred, and the requirements traceability will not be complete until the design of the final product.

(c) An incremental approach may have a higher risk of changes in requirements being identified during development because the delivered versions are subject to operational use at an early stage. This is seen as a particular problem with 2167A because of the amount of documentation which will need to be changed.

5.1.2 Evolutionary development

An evolutionary development approach is similar to incremental development; however, the interim deliverables are not meant to be final. The final product is built by slightly changing system characteristics through a successive number of increments during which the users' requirements are more fully defined.

The main concern with this method is that it assumes that the requirements and the design will change during the course of development. Because 2167A specifies a strict and detailed hierarchical documentation of the analysis and design process, it is feared that attempting to maintain full 2167A documentation of the process will be prohibitively costly and that the effort required to maintain documentation will make evolutionary development unfeasible in any reasonable timeframe.

5.1.3 Spiral model

The spiral model is a risk-driven approach based on a sequence of progressive cycles. Each cycle consists of four steps. At the end of each cycle, a review takes place to assess risks and to define activities for the next cycle. Because of its broad definition, the spiral model is a generic model which can encompass a number of other approaches (such as the evolutionary and waterfall models).

A common misconception is that the spiral model is based on an evolutionary or rapid prototyping approach. The problems of using 2167A with these approaches are often reported as a problem with the spiral model, whereas the problems are in reality associated with an instance (or instances) of the model.

5.1.4 Object oriented (OO) analysis and design

There are several object oriented developments being conducted in Australia in accordance with 2167A. Although the SRS now refines requirements into "capabilities" to accommodate OO methods, most developers consider that further tailoring is essential to document the analysis and design.

Comments on using OO methods with 2167A included the following:

(a) With OO methodologies the analysis and design are less distinct, which can cause problems in deciding where to document - in the SRS or the SDD.

(b) The "bottom up" nature of OO design, particularly using class libraries, is difficult to document in the SDD.

(c) Handling a large hierarchy of object classes can be difficult in the format of the SDD.

Most developers saw OO methods, and OO design in particular, as an inevitable ingredient in future software system development. DOD-STD-2167A does not appear to cater for these methods effectively, and guidance on tailoring is required if a consistent approach to documentation is to be expected.

5.2 Requirements analysis and design confusion

One subject that was raised several times in interviews was confusion between the analysis and design activities and their documentation.

Several developers claimed that the SRS forces design to be documented in the requirements analysis stage of the project. It was also evident that bad experiences with 2167A invariably coincided with the inclusion of too much design detail in the SRS. Other developers consider that the confusion of analysis and design is a common problem for inexperienced analysts, and is particularly penalised in 2167A because of the separation of the descriptions of the two activities into different documents.

Some customers described SRS documentation which consisted almost totally of design information and included no requirements analysis at all.

5.3 Partitioning

There were conflicting opinions among contractors regarding the rules used to partition software into CSCIs, then into CSCs and CSUs.

Most developers believe that the number of CSCIs must be minimised, to reduce the number of individual documents and inter-CSCI interfaces. Several had had unpleasant experiences with projects where too many CSCIs had been defined resulting in over-documentation and over-detailed formal testing. There was common agreement that software should form a single CSCI except where:

- Different parts of the software are being developed at different sites or by different contractors.

- The size of the CSCI would be unmanageable.

- Separate elements of the software run on different platforms or perform distinctly different functions.

Customers also showed concern about there being too many CSCIs, but some cited examples where all the software for a project was proposed as one CSCI, even though different programs ran on different platforms. There is concern that visibility important to the customer is lost in such a case.

There was less agreement on the partitioning of a CSCI into CSCs and CSUs. Several developers considered that a CSU should represent an individual process, such as a procedure, function or task. Others, particularly those with genuine experiences with a 2167A project, suggested that a compilation unit in Ada, or an individual source file in other languages, is more appropriate. Those using object oriented methods tended to allocate a CSU to each object.

One problem indicated with the "process per CSU" approach is that each process should be documented to the standard required in the SDD, which can result in an unnecessary documentation blow-out. It can also cause serious configuration management problems if several CSUs are in the same source file.

The "source file per CSU" approach also presents problems, because the documentation requirements for a CSU in the SDD appear to indicate a single process, or at most a compilation unit with a single process point of entry. Some tailoring may be necessary to document a CSU containing several related utility processes, or which represents an object with several methods. Some developers also suggested that a CSU consisting of more than one source file, or several data files, would be difficult to manage from a configuration viewpoint.

6. PRODUCTION ISSUES

6.1 Internal product evaluations and testing

It was evident from the interviews that several developers did not fully understand the requirements of 2167A with regard to internal product evaluation, and did not follow a formal evaluation process particularly with regard to documentation. This may partly account for the confrontations which have occurred on document delivery for design reviews.

Smaller developers in particular also have difficulty in providing genuinely independent product evaluation and test functions, possibly resulting in a lower quality product. Their answer to this criticism was that a smaller team has more cohesion and motivation than a larger one, leading to a higher quality product overall.

6.2 Configuration management

Many developers were concerned that their configuration management tools and procedures were not totally adequate for medium to large projects. They also considered configuration management to be a major problem in their projects. Some of the arguments with regard to partitioning (Section 5.3) were based on limitations in their current tools and procedures, rather than an objective approach to the problem.

6.3 Software development files (SDFs)

Several customers regard the quality and contents of SDFs as very important, both as a vehicle for visibility in the development phase (particularly with regard to peripheral design issues and test coverage for software elements), and also for use in later software maintenance.

Developers were divided on their attitudes towards SDFs. Some have internal standards for the format, representation and content of SDFs, and include them in a configuration baseline. Others regard SDFs as being rather like design notebooks, with no formal significance. In general, developers saw no objection to providing visibility of their SDFs, but several were opposed to including them as project deliverables.

7. TAILORING ISSUES

7.1 Tailoring experiences

Most developers had conducted at least one tailoring of 2167A. In some cases the tailoring was for (possibly unsuccessful) contract tenders or internal developments.

Customer tailoring experience was rarer, and often gained from a joint undertaking with their selected developer. The authors saw only two cases where customers had suggested a particular tailoring as part of the Request for Tender (RFT) documentation.

All participants with experience in tailoring saw it as a difficult process which challenged their understanding of 2167A and the software engineering process in general. Those who did not regard tailoring as a serious issue in 2167A development had not experienced a 2167A project or attempted to tailor the standard.

Many participants regarded tailoring as a "whittling" of the requirements of 2167A and the DIDs, ie as the removal of requirements. While the formal tailoring guidelines restrict tailoring of the DIDs to the removal of clauses and documents, addition or replacement of clauses in the standard itself is not only possible but recommended in some cases (the so called "shell" requirements). The authors believe that similar tailorings may be necessary in the DIDs in some circumstances.

7.2   Tools and courses

7.2.1   MIL-HDBK-287

MIL-HDBK-287, "A tailoring guide for DOD-STD-2167A", provides advice and a step by step approach to tailoring 2167A. Most participants who had attempted to tailor 2167A were familiar with the handbook and had found it useful. There were several criticisms, however, that the handbook's solutions were too general, and did not provide advice on the detailed tailoring of 2167A or its DIDs.

It was suggested that the handbook was less useful for those who understand and have used 2167A, ie that it addressed the obvious problems but avoided more difficult ones.

It also restricts tailoring of the DIDs to deletion or partial deletion of requirements. Most participants considered that a responsible tailoring may require some modification of requirements in the DIDs, particularly for some software development methods.

The sections on relationships to other standards were considered to be particularly useful.

7.2.2   The TAILOR tools

Several developers and customers had used Logicon's TAILOR computer based tools in tailoring 2167A. These tools are available in Australia from Technology Australia and comprise the following:

TAILOR/2167A	Tailoring of the 2167A requirements.
TAILOR/DIDs	Tailoring of the 2167A DIDs.
INSIGHT/2167A	A training tool for 2167A.
CDRL-GEN	Generation of Contract Data Requirements Lists (CDRLs).

Most found the tools to be useful, particularly with the first tailoring and as learning tools. Participants with some experience of tailoring saw less value in the tools. The tools are strongly based on the recommendations of MIL-HDBK-287 and therefore follow the strict rules of tailoring referred to above (Section 7.2.1).

The general opinion was that the TAILOR tools are useful for broad tailoring and for documentation of the chosen tailoring. It was stressed by many participants that the tools are not a universal and simple solution for tailoring - the user must still have a thorough understanding of the purpose and applicability of 2167A. In one case a participant had repeated the tailoring process up to 20 times for a project using these tools, as his understanding of the tailoring process and its potential consequences increased.

In at least two of the projects the customer and developer used TAILOR together to reach agreement on tailoring. Customers and developers who had done this were very satisfied with the results of the exercise, not only because of confidence in the acceptability of the tailoring, but also because of the increased level of understanding of each other's perspectives.

The use of CDRL-GEN was seen as of less value in Australia, where few CDRLs are presented using the formats provided (DD Form 1423 and AF Form 585). A few participants considered this tool very useful for the compilation of the CDRL however, and one had converted the form-based output to a table after generation of the CDRL.

7.2.3 Tailoring courses

The only course directly addressing tailoring that participants had attended was that provided by Technology Australia. (Other courses on more general subjects such as an introduction to 2167A also address tailoring, but in less detail).

The perceived value of the 2-3 day Technology Australia course varied with the needs and experience of those attending, although all found some value in the course. Customers with little 2167A experience considered that it a provided a good to insight into 2167A and its tailoring (some rated it as excellent). More experienced attendees found that the introductory approach to 2167A was unnecessary. Several participants would have liked to have had more hands-on experience with the TAILOR tool which was used in the course.

A few inexperienced participants had hoped that the tailoring course would provide them with enough experience of 2167A and tailoring for them to immediately begin tailoring for a medium to large size project (ie to become an "instant expert"). Although they found the course to be valuable, they admitted that the experience required and problems faced were greater than they had anticipated.

8. CONCLUSIONS

The survey revealed that problems in defence software projects in Australia stem from four main causes:

- Inexperience of customers in software development and the use of DOD-STD-2167A.

- Inexperience of developers in software development and the use of DOD-STD-2167A.

- Insufficient interaction between customers and developers.

- A limited understanding and lack of agreement of the tailoring necessary in a particular project, leading to inadequate tailoring.

It should be stressed that these failings are not universal, and that developer inexperience, for example, may not be a factor in all of a particular developer's projects (due to the use of different teams). Problems due to these causes are common, however, and need urgent attention.

8.1 Customer inexperience

The problems caused by customer inexperience are seen as follows:

- A reluctance to accept reasonable tailorings of 2167A, and an inability to contribute to the tailoring process.

- A tendency to enforce literally standards which are not appropriate for the project.

- An inability to recognise defects in the developer's software development process, and suggest corrective action.

8.2 Developer inexperience

The problems caused by developer inexperience are seen as follows:

- Not using recognised development methods, particularly for analysis and design, even when the developer claims to use (and has written standards for) such methods. There is a tendency for developers to proceed directly to design without adequate analysis of requirements. This leads to difficulties in documentation of the analysis and design phases, particularly with regard to requirements traceability.

- Insufficient understanding of the purpose of DOD-STD-2167A, leading to difficulties in the development and maintenance of documentation. This is also seen as a document-driven approach to development.

- Insufficient knowledge and experience in the tailoring of DOD-STD-2167A, leading to unacceptable or inappropriate tailorings.

8.3 Customer/developer interaction

Customer/developer interaction, particularly in the early stages of a project, is very important in insuring a relatively smooth path through formal reviews. Many projects have suffered delays at PDR or CDR when customers have rejected designs and documentation. Apart from the delays, the effort in producing documentation which requires major changes to meet the customers requirements can be minimised if customers can be provided with more visibility of the analysis and design process.

8.4 Tailoring

The problems caused by incorrect or nonexistent tailoring are seen as follows:

- Excessive documentation effort required, particularly if no tailoring is agreed.

- Inadequate and inappropriate documentation, particularly if modern development methods are used.

- Resistance to change due to the amount of documentation which must reflect the change.

- A tendency for developers to avoid or pay only token regard to what they consider to be draconian requirements.

9. THE WAY AHEAD

The survey indicates that there is a necessity for the establishment of tailoring guidelines for DOD-STD-2167A projects in Australia. Current guidelines, tools and training courses, while useful in educating customers and developers alike, offer only a broad introduction to tailoring.

Guidelines for different types of projects should form a baseline from which customers and developers can negotiate a tailoring of DOD-STD-2167A for specific aspects of a project.

Such guidelines should cater for the different characteristics of a project, including:

- Size and complexity (including the number of CSCIs).

- Whether a project is an internal development (eg DSTO) or a procurement.

- Whether the project is for the development of a feasibility demonstration, prototype or production system.

- The development methodology to be used.

The guidelines should also address the tailoring of other standards, particularly MIL-STD-1521B, to ensure compatibility with DOD-STD-2167A.

THIS IS A BLANK PAGE

APPENDIX I

REQUEST FOR INTEREST

The following is an example of the request for interest sent to potential participants. These included policy, project and research areas in the Department of Defence, software developers in industry and selected academic institutions.

=================================

TAILORING DOD-STD-2167A FOR DEFENCE SOFTWARE PROJECTS

1.   DOD-STD-2167A, the US DOD standard for software development for defence systems, is now mandatory for most software systems developed for the Department of Defence. However, it has become apparent that there is some uncertainty both in the Department of Defence and in Defence Industry about the process of tailoring the requirements for DOD-STD-2167A projects.

2.   DOD-STD-2167A attempts to cover all possible scenarios and is meant to be tailored to the size and requirements of each individual project. MIL-HDBK-287 addresses the tailoring process in broad terms, but does not provide a suitable set of models for Australian Defence software projects, as would appear to be required. In small to medium sized projects ignoring the tailoring process could lead to overkill in documentation, review procedures and testing. Even in the largest projects, tailoring should produce significant savings in cost as well as producing a better end product.

3.   A DSTO research study is being conducted to investigate tailoring for internal software projects and Defence software procurement. The study is a joint undertaking of Combat Systems Division in Weapons Systems Research Laboratory and Information Technology Division in Electronics Research Laboratory. A small number of projects have already been carried out in Australia under DOD-STD-2167. A review of some of these projects, including the way they were undertaken, will be carried out as an initial part of the research. In addition, it is proposed to seek the assistance of the Australian Defence software community (both within the Department of Defence and in private contracting organisations) in order to study the types of tailoring that will be required for future Defence projects.

4.   Accordingly, letters similar to this one are being sent to all those organisations in the Australian Defence Community who may have an interest in supporting us in this study. We are also advising software engineering research groups within Australian Universities and Professional Societies of the existence of the study in case they wish to participate.

5.   The study is expected to take approximately twelve months to complete. The final report is expected to contain a guide to the types of tailorings of DOD-STD-2167A that are appropriate to possible Defence software projects, as well as recommendations as to how the tailorings can be implemented in the Defence contracting environment, for example by use of computer aided tailoring tools.

6.   We are now seeking expressions of interest from members of the Australian Defence community who are willing to assist us in this study, and participate in discussions on the use of DOD-STD-2167A. Suggestions as to the types of projects that should be considered, and notes of experiences (good and bad, verbal or written) with DOD-STD-2167, would also be useful at this early stage.

7.   If you feel that your organisation could contribute to or benefit from this study, please nominate a contact for further correspondence. Responses should be made to Peter Pollard by mail, phone ((08) 259 7083), fax ((08) 259 6781) or e-mail (pcp@csd0.dsto.oz.au).

J.M. WILSON
Chief

04 Oct 90

APPENDIX II

INTERVIEW QUESTIONNAIRE

This appendix contains the questionnaire used as a basis for discussion with participants in the survey. Some questions were deliberately intended to be naive and/or provocative to stimulate discussion.

=================================

The following questions will form the basis for our discussions. They are provided in advance to allow you to consider the various issues.

We do not intend to include details of these interviews in any of our subsequent reports, and would prefer the discussions to be as informal as possible. All information provided will be considered as confidential.

Background

(1)   Types of project (eg size, number of CSCIs etc) - (if not DOD-STD-2167A, experience with other earlier standards, especially DOD-STD-2167, may be valuable). What languages did you use for each project?

(2)   What Tailoring if any was applied and what tailoring aids did you use, eg MIL-HDBK-287, Logicon?

(3)   What mistakes, misconceptions, traps were experienced and what was the pain?

(4)   What software development methodologies and CASE tools have you used and how compatible were they with DOD-STD-2167A?

(5)   How do you view the interaction of 2167A with other standards (eg 490, 1521, Australian standards)? Are there serious overlaps or conflicts?

(6)   How did you find the hardware design cycle (if any) interacted with the software design cycle? Was 2167A a help or a hindrance?

(7)   Do you have any suggestions as to how the customer can guarantee good documentation (using any standard)?

(8)   Do you have written software development and coding guidelines?

(9)   How well do you believe your staff understand the requirements and reasons for using 2167A? What relevant training is provided for your staff? How well do you believe your customer/contractor understands 2167A?

Tailoring

(10) how do you feel that the tailoring should be affected by unique project factors such as size, real time performance, number of CSCIs, time scale, cost, safety critical, etc? What factors do you believe are most critical with regard to tailoring.

(11) Under what circumstances do you believe that formal reviews should be omitted or merged (eg combining of PDR and CDR)?

(12) How do you partition software into CSCIs? Do you have any comments on the result of having too many or too few CSCIs? Do you have similar comments with respect to CSCs?

(13) It has been suggested that the 2167A (and 490A) standards are unsuitable for the definition of operational and functional requirements because they inhibit the user's (operator's) ability to understand and hence contribute to the requirement. Do you have any comments on this?

(14) Design/requirements tradeoffs - does the SRS force design during the requirements phase?

(15) What are your views on the relative values of SRS/IRS, and SDD/IDD documents, especially for small projects?

(16) Do you regard requirements traceability as important? When could it be relaxed?

(17) How do feel about the early definition of qualification requirements and methods, including FQT?

(18) What methods do you use for product evaluation (eg peer review)? How important is product evaluation in your opinion?

(19) Do you agree with the requirements for configuration management under 2167A? Under what circumstances would you consider changing these?

(20) Do you have any comments with respect to transitioning from development to support under 2167A?

(21) How importantly do you regard SDFs? In what form do you maintain SDFs?

(22) Some feel that tailoring will always result in a loss of visibility and there is therefore a natural reluctance to tailor. How do you answer this argument?

(23) Do you feel that a contractor benefits in any way when obliged to use 2167A?

(24) Do you feel that 2167A lends itself to change (ECPs) during the development cycle? What do you consider to be the advantages and disadvantages of 2167A in this regard?

(25) How difficult is it to do OOD and OOP under 2167A. Is this a serious problem given the way software engineering is heading?

(26)   Have you considered the use of 2167A for other than the traditional life-cycle
        models?

DISTRIBUTION

Defence Science and Technology Organisation

Chief Defence Scientist and members of the DSTO Central Office Executive	1
Counsellor, Defence Science. London	Cnt Sht Only
Counsellor, Defence Science, Washington	Cnt Sht Only
Scientific Adviser to Defence Central	1
DSR - Bangkok	Cnt Sht Only
SA to DRC - Kuala Lumpur	Cnt Sht Only

Weapons Systems Research Laboratory

Director, Weapons Systems Research Laboratory	1
Chief, Combat Systems Division	1
Head, Combat Systems Integration	1
A.P. Gabb, Combat Systems Integration	5
P.C. Pollard, Combat Systems Integration	60

Aeronautical Research Laboratory

Director, Aeronautical Research Laboratory	1

Electronics Research Laboratory

Director, Electronics Research Laboratory	1
Chief, Information Technology Division	1
Head, Software Engineering	6
R.J. Vernik, Software Engineering	4

Surveillance Research Laboratory

Director, Surveillance Research Laboratory	1

Materials Research Laboratory

    Director, Materials Research Laboratory   1

Libraries and Information Services

Librarian, Technical Reports Centre, Defence Central Library, Campbell Park 1

Manager, Document Exchange Centre   1

    United Kingdom, Defence Research Information Centre   2
    United States, National Technical Information Service   2
    Canada, Director Scientific Information Services   1
    New Zealand, Ministry of Defence   1
    National Library of Australia   1

Main Library, Defence Science and Technology
  Organisation Salisbury   2
Library DSD Sydney   1
Library DSTO Sydney   1
Library MRL   1
Library EDE   1

Australian Defence Force Academy Library   1

British Library Document Supply Centre   1

The Technical Cooperation Program (TTCP)

Australian National Leader, Subgroup X   1

Department of Defence

Director of Departmental Publications for AGPS   1

Scientific Adviser to Defence Intelligence Organisation   1

Navy Office

    Navy Scientific Adviser   1

    Director, Navy Combat System Engineering   1

Army Office

    Scientific Adviser - Army   1

    Director, Command and Control Procurement, Army   1

Air Office

Air Force Scientific Adviser	1
Director, Electronics Engineering, Air Force	1
Spares	10
Total number of copies	121

DOCUMENT CONTROL DATA SHEET

Security classification of this page : UNCLASSIFIED

1 DOCUMENT NUMBERS	**2** SECURITY CLASSIFICATION
AR Number : AR-006-788	a. Complete Document : Unclassified
	b. Title in Isolation : Unclassified
Series Number : WSRL-TN-57/91	c. Summary in Isolation : Unclassified
Other Numbers :	**3** DOWNGRADING / DELIMITING INSTRUCTIONS

4 TITLE

TAILORING DOD-STD-2167A - A SURVEY OF CURRENT USAGE

5 PERSONAL AUTHOR (S)	**6** DOCUMENT DATE
A.P. Gabb, P.C. Pollard, S.F. Landherr and R.J. Vernik	October 1991
	7 7.1 TOTAL NUMBER OF PAGES 25
	7.2 NUMBER OF REFERENCES -

8 8.1 CORPORATE AUTHOR (S)	**9** REFERENCE NUMBERS
Weapons Systems Research Laboratory	a. Task : DST 89/218
	b. Sponsoring Agency :
8.2 DOCUMENT SERIES and NUMBER Technical Note 57/91	**10** COST CODE
11 IMPRINT (Publishing organisation)	**12** COMPUTER PROGRAM (S) (Title (s) and language (s))
Defence Science and Technology Organisation	

13 RELEASE LIMITATIONS (o. the document)

Approved for Public Release.

Security classification of this page : UNCLASSIFIED

14 ANNOUNCEMENT LIMITATIONS (of the information on these pages)

No limitations

15 DESCRIPTORS		**16** COSATI CODES
a. EJC Thesaurus Terms	Computer programs Military standards Defense programs Department of Defence (Australia)	1205, 1506
b. Non - Thesaurus Terms		

17 SUMMARY OR ABSTRACT
(if this is security classified, the announcement of this report will be similarly classified)

(U) As part of a study into the tailoring of DOD-STD-2167A, a survey was conducted into the use and tailoring of the standard in Australia. This paper summarises the results of the survey and makes recommendations for the development of tailoring guidelines.